FTPM Log

Diagnostic for FTPMAN protocol

Mon, Sep 22, 2003

The Acnet RETDAT protocol writes short records of the requests it receives into a data stream called RETDLOG. This note describes a similar scheme for writing records about requests received in the FTPMAN protocol.

Consider the following 16-byte structure to hold a record of an FTPMAN request:

Field	*Size*	*Meaning*
rqNode	2	Requesting node#
rqType	1	FTPMAN request type code
rqNDev	1	#devices
rqNPts	2	#pts/device for snapshot, #pts/reply/device for continuous
rqDTime	2	Elapsed time to process request, in microseconds
rqMsgId	2	Message-id from Acnet header
rqTime	6	Time of day, as Da,Hr,Mn,Sc,Cy,Ms.

One can see it is a tight squeeze to fit all this into 16 bytes, for convenience in working via the usual memory dump page. The rqNode field is specified as the Acnet node#, since all such requests include one in the Acnet header. (Actually, that node field will be seen as a pseudo node#, but using GetIPARP, it can be used to get the node number field in the indicated IPARP table entry, as in 0x0B21, for example.)

The type code is part of the FTPMAN request message header. Values supported are 1=timing, 5=snapMisc, 6=continuous, 7=snapShot, and 8=snapData. The #devices applies only for types 1, 6, 7. For a server request, the elapsed time covers the time spent initializing the server request. For a nonserver request, it may be that the request is simply ignored because it does not see its own node# represented in the various SSDNs, in which case no log entry is written.

The time-of-day is only 6 bytes, in order to make room for the message id. If the error code is negative, the Day byte of the time is overwritten. In this way, the day information is only lost in case of a returned error status. Pending error status (positive values) are not reported.

This diagnostic only logs requests, not replies. Use other tools to monitor replies. But it logs cancels, too, since that will show when a message-id has been retired. For a cancel, the fields rqType, rqNDev, rqNPts are zero.

The detailed implementation involves two functions, FTPMLogI and FTPMLogF. The calling sequences are:

```
PROCEDURE FTPMLogI(rqNod, rqMsId, rqTy: Integer);
PROCEDURE FTPMLogF(nDv, nPt, rqEr: Integer);
```

Several globals are also used for convenience:

```
diagTStart: Longint;  { microsecond counter at start of processing }
diagNPts: Integer;    { diagnostic #points requested }
diagNDev: Integer;    { diagnostic #devices requested }
diagQR: DiagDSRType;  { diagnostic record format }
```

The FTPMLogI routine is called at the start of processing a request. The FTPMLogF is called when finishing processing, which causes the record to be written into the data stream.